

Solaris (TM)

USB Dual Framework Guide

Copyright 2003 Sun Microsystems, Inc. 901 San Antonio Road, Palo Alto, California 94303 U.S.A. All rights reserved.

This document and the product to which it pertains is distributed under licenses restricting its use, copying, distribution, and decompilation. No part of the product or of this document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Solaris, Sun Blade, The Network is the Computer are trademarks, registered trademarks, or service marks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

Federal Acquisitions: Commercial Software -- Government Users Subject to Standard License Terms and Conditions.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.
Copyright 2003 Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, California 94303 Etats-Unis. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, Solaris, Sun Blade, The Network is the Computer, sont des marques de fabrique ou des marques déposées, ou marques de service, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REpondre A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Solaris (TM) USB Dual Framework Guide

I. <u>Intro to USB</u>	4
1. <u>What is USB</u>	4
2. <u>Sun systems coming with USB</u>	4
3. <u>How to add USB or USB 2.0 hardware to your system</u>	4
4. <u>Dual Framework Software Installation</u>	5
II. <u>Dual Framework</u>	6
1. <u>What is the Dual Framework</u>	6
2. <u>Configurations</u>	6
3. <u>How to Tell Existing Configuration</u>	8
4. <u>How USBA Configuration Works</u>	9
5. <u>Configuration and System Upgrades</u>	9
6. <u>usba10 Property</u>	10
7. <u>Driver Compatibility</u>	10
8. <u>Purposefully Disabling Specific Types of USB Devices</u>	10
9. <u>Use of USB Keyboards and Mice on Legacy Systems</u>	11
10. <u>Manpages</u>	13
III. <u>USBA 1.0 Framework</u>	13
1. <u>Features</u>	13
2. <u>Limitations</u>	14
IV. <u>All USB Storage Devices Treated as Removable Media</u>	16
1. <u>What This Change Means</u>	16
2. <u>Caveats</u>	16
V. <u>USB Floppy Devices</u>	17
1. <u>Use with Volume Manager and File Manager</u>	17
2. <u>Manual Configuration</u>	17
VI. <u>Driver Development Kit</u>	18

Solaris (TM) USB Dual Framework Guide

This guide describes the Universal Serial Bus (USB) Dual Framework. The intended audience includes USB administrators and users of SPARC systems running Solaris(TM) 8 Operating System HW 5/03 or the Solaris 8 OS release with patch 109896 rev 19 or later, and users of a USB driver development kit (DDK).

This document is an update to the System Administration Guide. Defer to this document when there is conflicting information between the System Administration Guide and this document. This document also provides more detailed release information.

I. Intro to USB

1. What is USB

USB, or Universal Serial Bus, is a hardware and software specification which supports certain devices which can be connected or disconnected to your computer at any time. Devices are automatically detected and drivers are auto-configured for immediate use of the device.

USB devices are of many types. Newer PCs and Sun equipment feature USB keyboards and mice. There are also mass storage, printer, audio and many other kinds of devices.

USB devices fall into three speed categories. The older USB 1.1 specification defines low and full speed devices, which operate at 1.5Mb/sec (megabits/sec) and 12 Mb/sec (maximum theoretical bandwidths) respectively. (This document refers to these as USB 1.1 devices.) Most keyboards and mice fall into this classification, as well as older USB devices. The newer USB 2.0 specification defines high speed devices, which operate at 480Mb/sec. This document refers to these as USB 2.0 devices.

Trees of many USB devices can be connected through external hubs, which split out one USB connection into several. The USB connections on the computer itself are made available by root hubs. The USB specification allows for up to 127 devices to be connected per bus. (This Solaris OS release supports up to 126 devices per bus.)

2. Sun Systems Coming with USB

Sun has supported USB on the desktop since the introduction of the Sun Blade (TM) 100 in 2000. Current systems include the Sun Blade 100 and 150, and Sun Blade 1000 and 2000 workstations. All of these systems support the USB 1.1 specification on their motherboard, which means they can operate USB 1.1 devices at their full speed, and can operate USB 2.0 devices at only at maximum USB 1.1 device speed (12 Mb/sec) when connected through motherboard ports.

3. How to Add USB or USB 2.0 Hardware to Your System

Any PCI-based sun4u architecture system can accommodate after-market PCI-based cards offering USB 2.0 support, enabling high speed use of USB 2.0 devices. Please see <http://www.sun.com/io> for a list of Solaris OS Verified PCI USB 2.0 cards. USB 2.0 support in the dual framework configuration is first made available as Solaris 8 OS HW 5/03 and as Solaris

8 OS patch 109896-19. Prior versions of Solaris 8 OS will operate any USB 2.0 ports as USB 1.1 ports. Please see the configuration section for further information on how to use and configure USB 2.0 ports.

4. Dual Framework Software Installation

There are four ways of installing the Dual Framework on a system running a Solaris 8 release:

- A. Install the Solaris 8 HW 05/03 or later release. This is the easiest (and preferred) way to get the Dual Framework as well as all other libraries and support programs to complete the USB environment.
- B. Run patchpro (available under “Basic Analysis Tools” at sunsolve.sun.com) to find the dependency tree of patches required for the USB Dual Framework patch (109896-19 or later).

Patchpro requires installation, which is discussed in its README. Note that the “setup” command phase can take several minutes to run without displaying any error messages, so be patient. Once installed, manpages for patchpro are available at `/opt/SUNWppro/man`.

Once installed, use patchpro to find dependencies for the USB Dual Framework patch. If the latest version of the patch is 109896-19, run patchpro as:

```
/usr/sadm/bin/smpatch download -i 109896-19
```

Whether or not patchpro terminates abnormally (because it may not be able to get the patches), the command will first display a list of patch dependencies. For 109896-19, relevant output is as follows:

```
The following patches were added due to patch dependencies:  
112396-02  
108987-13  
111111-03  
111310-01  
108528-22  
109883-02  
110609-04
```

Next, download all required patches (via patchpro, manually at sunsolve.sun.com, or otherwise) if not already downloaded, unzip and install (via patchpro, `patchadd(1M)`, or otherwise). The all patches can be installed together, if they are downloaded and unzipped within a single directory. First, install dependency patches in the order listed in the patchpro output, and then install the USB Dual Framework patch afterward, for example:

```
# cd <dir-where-unzipped-patches-are>  
# patchadd -M . 112396-02 108987-13 111111-03 111310-01 \  
108528-22 109883-02 110609-04 109896-19
```

Note that if you want full functionality of all that USB supports, you must install additional

patches. Please check the “Special Install Instructions” section of the USB patch README for a list. Patchpro can be used to track the dependencies of these patches the same way as is shown above for the USB Dual Framework patch.

C. Install the Solaris 8 Recommended Solaris Patch Cluster, which includes the Dual Framework patch and other required patches. This type of installation eliminates the need for patchpro, but downloads many patches not required for USB (total of ~90 Mb). As with option (B) above, there may be additional patches needed to support full functionality of all that USB supports. See the “Special Install Instructions section of the USB patch README for a list, and then verify (and install if necessary) these patches and their dependencies. “showrev -p” can be used to display installed patches and their versions.

D. Install the USB DDK (Driver Development Kit). See section VI for more information on obtaining a DDK. This installation type is good for developing a driver which will run on the USBA 1.0 Framework. Install any desired patches before installing the DDK. Note that performing a system OS upgrade on a system with a DDK installed is not supported and may fail.

II. Dual Framework

1. What is the Dual Framework

The Solaris 8 OS USB Dual Framework is a service layer that enables USB devices in the Solaris OS. The "original" framework, found in Solaris 8 OS 2/02, was developed with USB 1.1 devices in mind. With the advent of higher-speed USB 2.0, a new framework, called USBA 1.0, was created to meet more demanding requirements of USB 2.0 devices. To facilitate smooth transition from the original framework to the newer one, Solaris 8 OS provides both, which can coexist on the system simultaneously, hence the name Dual Framework. Note that the original framework will be removed in a future Solaris OS release.

The original framework fully supports USB 1.0 and USB 1.1 devices, and supports USB 2.0 devices as if they were USB 1.1 devices (and at USB 1.1 speeds). By default the original framework operates only devices connected to USB 1.0 or 1.1 ports. Older, non-Sun drivers will work only with this framework.

By default, the USBA 1.0 framework operates devices connected to the computer's USB 2.0 ports. Sun systems are not shipped with USB 2.0 hardware, but see section I.3 regarding adding Solaris OS Verified USB 2.0 hardware.

Another enhancement to USB with this Dual Framework release is more robust mass storage support. This is available with both frameworks. All USB mass storage devices are seen as removable, eliminating confusion around prior differing behavior among USB mass storage devices. Please see section IV for more information on mass storage specifics.

2. Configurations

The configuration of the two frameworks can be altered. The different configurations are shown in the table 1 below, along with their merits and issues. Please see section II.4 for instructions on how to change configuration. Please note that only the default configuration is supported by

Sun for this release.

Configuration	Pros and Cons
2.0 ports / USBA 1.0 1.x ports / Original Framework (default)	<ul style="list-style-type: none"> • Pros: <ul style="list-style-type: none"> • USBA 1.0 framework is available for USB 2.0 devices • Original framework is available for compatibility with older non-Sun drivers • Cons: <ul style="list-style-type: none"> • Userland driver support will be available only through the system's* USB 2.0 ports. • Unless a device or device class has driver versions for both frameworks, that device or class will not work on all ports. • Mass storage devices connected through a system's USB 2.0 port will not be readily seen by vold upon reboot.** • No USB 1.x devices can be used behind an external USB 2.0 hub connected to a system's USB 2.0 port.
All ports / USBA 1.0	<ul style="list-style-type: none"> • Pros: <ul style="list-style-type: none"> • Uniform behavior on all ports. • All ports are operated with the more evolved framework. • Cons: <ul style="list-style-type: none"> • Older non-Sun USB drivers will not work. • No USB 1.x devices can be used behind an external USB 2.0 hub connected through a system's USB 2.0 port.
All ports / Original Framework	<ul style="list-style-type: none"> • Pros: <ul style="list-style-type: none"> • Uniform behavior on all ports. • Older non-Sun USB drivers will work on any port. • All devices will work behind any external 1.x or 2.0 hub. • Cons: <ul style="list-style-type: none"> • USB 2.0 devices will function only as 1.1 devices, at slower 1.1 speeds, even when connected to USB 2.0 ports. • There will be no userland driver support for devices on any port. • Drivers written only for the USBA 1.0 framework (such as UGEN) will not work.
Original Framework ohci driver USBA 1.0 ehci driver	<ul style="list-style-type: none"> • Pros: <ul style="list-style-type: none"> • Older non-Sun USB drivers will work on any port. • Uniform behavior on all ports (except as noted below). • Cons: <ul style="list-style-type: none"> • All USB 1.x devices will operate with the less evolved original framework. • No USB 1.x devices can be used behind an external USB 2.0 hub plugged into a USB 2.0 port.

* A system USB port is a port on the computer itself, sometimes called a root hub port.

**Vold sees mass storage devices after a reboot, under one of the following conditions:

- all devices, connected through any computer USB port, are found after reconfigure reboot (boot -r)
- devices connected though USB 1.1 computer ports, are found after all reboots

4. How USBA Configuration Works

This section describes device <-> driver bindings and is added background information for those who are interested. The recommended way of configuring the Dual Framework is with the “usbconfig” configuration script, found at www.sun.com/bigadmin/scripts/indexConfig.html.

All USB 2.0 cards contain both OHCI (Open Host Controller Interface) hardware and EHCI (Enhanced Host Controller Interface) hardware which work together. USB 1.x cards contain only OHCI hardware. OHCI and EHCI hardware are seen by this Solaris OS release as two devices, even though they may work together on the same card. The USB configuration can be changed based on the bindings of the host controller drivers (usba10_ohci, usba10_ehci, ohci) to these devices.

usba10_ohci is the USBA 1.0 framework host controller driver for low and full speed USB hardware support. In the default configuration, usba10_ohci is bound to devices which specify "pci1033,35" (1033=NEC, 35=USB), restricting the device to USB cards with an NEC controller on them. This binding is evident by grepping through /etc/driver_aliases for usba10_ohci.

usba10_ehci is the USBA 1.0 framework host controller driver managing high speed USB hardware support. In the default configuration, usba10_ehci is bound to devices which specify "pciclass,0c0320", or all devices in the PCI class of 0c0320 (or high speed USB host controllers).

OHCI is the original framework host controller driver, and is bound to devices which specify "pciclass,0c0310", or all low and full speed USB host controllers. This Solaris OS release will bind devices to the most specific drivers first, so this OHCI binding is a catch-all for those devices which didn't get bound to usba10_ohci.

These bindings have been selected since NEC is the dominant vendor of USB 2.0 chips, so USB 2.0 cards will be bound to both usba10_ohci and usba10_ehci. Devices attached to ports managed by these drivers will be operated by USBA 1.0 framework drivers.

(NOTE: These are the bindings delivered with Solaris 8 OS HW 5/03. These bindings could change in subsequent releases.)

Knowing this, the configuration can be changed so that all USB hardware is operated by the USBA 1.0 framework. Simply remove the ohci binding, and bind the usba10_ohci to the full class of "pciclass,0c0310" devices. The easiest way to do this is with the available “usbconfig” configuration script, available at www.sun.com/bigadmin/scripts/indexConfig.html.

5. Configurations and System Upgrades

System upgrades are supported only in the default configuration, and only when the Dual Framework has been installed as part of an OS release or an official patch. If the configuration has been changed, whether via the “usbconfig” script or otherwise, **the default configuration must be restored before performing an OS upgrade**. The easiest way to restore the default configuration is to use the “usbconfig” script, which has a selection for default configuration. **Systems which have the Dual Framework through a DDK installation should be fully**

reinstalled before being upgraded. Not meeting these conditions can cause the upgrade to fail.

6. usba10 Property

Several USBA 1.0 client drivers are re-implementations of “original framework” drivers, and co-exist alongside them in the dual framework. These client drivers have a `usba10` property defined in their `.conf` file. This is a Boolean property. If present, this property simply tells both frameworks that the driver is meant for the USBA 1.0 framework. The absence of this property tells both frameworks that the driver is meant for the original framework. Note that this property, plus any `.conf` files containing only this property, are planned to be dropped in a future release.

7. Driver Compatibility

When a device is plugged into a port operated by a framework that does not recognize a proper driver for that device, as when a framework tries to attach an incompatible driver or cannot find a compatible driver, then a message similar to the following will appear on the console:

```
The driver for <device binding name> is not for USBA1.0
```

This message will appear, for example, when a device driven by a non-Sun, USBA 1.0 driver, is plugged into a port supported by the original framework in the default configuration.

8. Purposefully Disabling Specific Types of USB Devices

Specific types of USB devices can be disabled by disabling their client driver. For example, USB printers can be disabled by disabling the `usbprn` driver which operates them. Disabling `usbprn` does not affect other kinds of devices, such as USB storage devices. Device types are disabled on both frameworks; there is no way to disable device types on one framework only. The following drivers are used to specify the following usb device types:

<i>Device Type</i>	<i>Driver</i>
audio	usb_ac and usb_as
hid (usually kbd, mouse)	hid
storage	scsa2usb and usb_sd
printer	usbprn
serial	usbser_edge

Disable a driver with the `update_drv(1M)` command as follows:

A. Save a copy of `/etc/driver_aliases` for a record of the alias(es) you are about to delete.

```
# cp /etc/driver_aliases /etc/driver_aliases.orig
```

B. Look in `/etc/driver_aliases` for the proper driver aliases to match the driver:

```
# grep usbprn /etc/driver_aliases
usbprn "usbif,class7.1.1"
usbprn "usbif,class7.1.2"
```

C. Issue the `update_drv` command with that driver and aliases:

```
# update_drv -d -i '"usbif,class7.1.1"' usbprn
# update_drv -d -i '"usbif,class7.1.2"' usbprn
```

D. Reboot the system.

Note that if a device of a disabled device type is connected, messages on the console similar to the following, may appear:

```
usbal0: WARNING: usba: no driver found for device <name>
```

9. Use of USB Keyboards and Mice on Legacy Systems

Setup of USB console keyboard and mouse on systems designed for use with them is automatic, but what about legacy systems?

USB keyboards and mice can be used as X-server input devices on PCI legacy systems delivered without USB support (assuming, of course, that a PCI card is installed with USB ports). They can also be used as X-server input devices on USB systems when connected to USB ports other than the on-board USB 1.1 ports. In order to do this, the console subsystem needs to be told where the keyboard and mouse are, via entries in the `/etc/system` file. Here is how to set this up:

A. Determine the USB ports to be assigned to the console keyboard and mouse.

B. Plug in the USB keyboard and mouse.

C. Run the following command to find the keyboard path:

```
ls -l /dev/usb | grep keyboard | nawk '{ print $11 }'
```

Output will look similar to this:

```
../../../../devices/pci@1f,0/pci@5/pci@0/usb@8,1/keyboard@3:keyboard
```

D. Strip off the portion to the left of the first `/pci`, to get this:

```
/pci@1f,0/pci@5/pci@0/usb@8,1/keyboard@3:keyboard
```

E. Strip off the `:keyboard` at the far right, to get this:

```
/pci@1f,0/pci@5/pci@0/usb@8,1/keyboard@3
```

This is the device path to use.

F. Repeat steps (C)-(E) for the mouse. `"mouse"` will appear wherever `"keyboard"` does. The

result will be similar to the following:

```
/pci@1f,0/pci@5/pci@0/usb@8/mouse@1
```

G. Using the paths just found, add lines to the `/etc/system` file similar to the following:

```
set consconfig:usb_kb_path="/pci@1f,0/pci@5/pci@0/usb@8,1/keyboard@3"  
set consconfig:usb_ms_path="/pci@1f,0/pci@5/pci@0/usb@8/mouse@1"
```

H. Reboot.

There are some caveats to this:

- This can be changed in a future Solaris OS release.
- The ports assigned to the keyboard and mouse by the `consconfig` entries above are to be used only for the keyboard and mouse, and must be the only ports used for them unless the paths are changed in the `/etc/system` file.
- Console keyboard and mouse must be connected directly to USB ports on the computer, not through an external hub.
- If possible, use a terminal connected to the `ttya` serial port, as the OBP console device. This allows full control over bootup, On-Board Prom (OBP) commands and variable manipulation, and the ability to issue `STOP` key commands (such as `STOP-A`) to the system. Please see the *OpenBoot 4.x Command Reference Manual* for more information on setting (what it calls) the console input and output device.

Note that what this document is calling the OBP console device is not necessarily the same as an X-server input device. While the two terms refer to the same thing in a standard configuration, here the OBP console and X-server input devices are different. The OBP OK prompt appears on the serial port OBP console device, while the X-server input device facilitates a CDE or windowing session on a graphic monitor.

- If a `ttya` OBP console is not possible, a legacy keyboard and mouse (or USB keyboard and mouse plugged into their standard USB on-board ports on USB systems) must be used for communicating with the OBP when needed. Reconnect them when you need to use them, as in the case of running diagnostics or manually booting or `fscking` the system. Take care to poweroff the system before connecting legacy keyboards and mice. Note that the `eeeprom` command can be used to change OBP settings while the system is up, and setting the OBP `auto-boot?` setting to `true` will eliminate the need to communicate through OBP for almost all cases except for grossly unclean shutdowns.
- A USB keyboard and mouse can be used as an X-server input device while connected to any USB port, without the need for an entry in the `/etc/system` file, if the delay in `/etc/init.d/initusb` is reduced. Please see section [III.2](#) for more information.

10.Manpages

For the Solaris 8 OS HW 5/03 release, the manpages covering the USB dual framework are delivered in a package separate from other manpages (SUNWs8hwman1 vs SUNWman), and install in a different directory tree than the other manpages (/opt/SUNWs8hwman1 vs /usr/man).

The USB manpages in /usr/man are part of the Solaris 8 OS 2/02 documentation set, and as such cover only the original framework. The scsa2usb(7D) and ohci(7D) manpages of the original documentation set are out of date. The USB manpages in /opt/SUNWs8hwman1 cover both the USBA 1.0 and original frameworks, and are current.

There are three ways of making the manpages in /opt/SUNWs8hwman1 accessible:

A. Copy them from the /opt/SUNWs8hwman1 directory tree to the /usr/man tree:

```
cp -r /opt/SUNWs8hwman1 /usr/man
```

Note that some of the original manpages in /usr/man will be overwritten.

B. Specify the /opt/SUNWs8hwman1 directory tree in the MANPATH before /usr/man:

```
ksh:      export MANPATH=/opt/SUNWs8hwman1:$MANPATH
csh:      setenv MANPATH /opt/SUNWs8hwman1:$MANPATH
sh:       MANPATH=/opt/SUNWs8hwman1:$MANPATH ; export MANPATH
```

C. Specify the /opt/SUNWs8hwman1 directory tree through the -M option to man(1):

Example:

```
man -M /opt/SUNWs8hwman1 usba
```

This command will look in the /opt/SUNWs8hwman1 manpage tree when searching for the usba manpage.

Note that this method is only temporary, whereas the first two methods are permanent.

III.USBA 1.0 Framework

1. Features

- USB 2.0 support

USB 2.0 devices follow the "high speed" protocol, allowing speeds up to 40X faster than their rival USB 1.1 devices. Devices need only be connected through a port on a USB 2.0 card driven by the USBA 1.0 framework, to be operated with the "high speed" protocol. USB 2.0 devices today include most mass storage devices, including CD-rom and DVD

burners, and hard drives.

- More robust framework

The USBA 1.0 framework is more evolved than the "original" framework, and as such offers better performance. Any USB device, including USB 1.x devices, connected to a port driven by this framework can take advantage of this.

- More drivers (will be) available for this framework

The USBA 1.0 framework includes changes to simplify and enable support of more kinds of devices. Documentation on how to write a USBA 1.0 framework driver is available to the public. This means that ultimately more drivers will be available for the USBA 1.0 framework than the original framework.

- Driver Development Kit available

Developers of kernel drivers can request full kernel documentation as part of a Driver Development Kit (DDK). The same DDK also gives documentation and headers for UGEN, the pass-through driver which allows raw USB device access from userland. Please see the end of this document for more information on how to get a DDK.

2. Limitations

A. Configuration

- USB keyboard and mouse can be recognized and used as console devices only on systems which support this configuration: Sun Blade 100, 150, 1000, 2000. On these systems, the console devices must be plugged into on-board* USB ports, because these ports have special software associated with them to recognize a USB keyboard and mouse connected through them as console devices. No systems at this time recognize a USB keyboard or mouse plugged in through a USB PCI card as a console device. Only recognized console devices can be used to boot the system and allow the console windowing system to start.

* On-board ports are USB 1.1 ports which come with the basic system, and are not resident on PCI cards.

- Connect USB 2.0 storage devices to the USB 2.0 combo card for best performance. While such devices will function on other ports, they will operate significantly slower, as USB 1.1 devices.
- USB 1.x devices will not function when connected through a USB 2.0 hub connected to a USB 2.0 port, due to a current software limitation. Either connect the USB 2.0 hub to an onboard or other USB 1.1 port, or bypass the hub and plug the USB 1.x device directly into a USB port on the computer.
- USB 2.0 isochronous transfers are not supported.

B. Delay when booting with non-configure boot

Some USB devices will not be available until approximately two minutes after a non-configure system boot completes. Devices affected are those operated by the USB 1.0 framework, and devices operated by the original framework on systems without a console keyboard and mouse.

This can be changed, and devices made available at login time, by editing the `/etc/init.d/initusb` script, and changing the "sleep 60" to "sleep 5". Booting the system will take a few seconds longer, as a side effect of making this change.

NOTE: Be very careful when making changes to this script, and make only the change prescribed above. Changing this script in other ways can have unknown side effects.

NOTE: This change works for modest-sized USB device trees. Each device in a tree requires time for enumeration, so there may not be enough time for many devices to enumerate even with this script change.

C. Limitations with Vold

Due to the delay of USB devices coming online after a non-configure boot, vold may not see USB mass storage devices automatically upon a reboot. This is because vold is started before the USB devices come online.

Choose one of the following workarounds:

- Reduce the delay (see (B) above). This is a one-time change which holds across reboots.
- Hotplug the needed USB storage devices approximately two minutes after the system has given the login prompt. This will need to be done for each USB storage device, after every reboot.
- As root, stop and restart vold:

```
/etc/init.d/volmgt stop  
/etc/init.d/volmgt start
```

Do this after every reboot, approximately two minutes after the system has given the login prompt.

Refer to the System Administration Guide and `scsa2usb(7D)` for more information.

- As root, send vold a SIGHUP signal to initiate a rescan for devices:

```
pkill -HUP vold
```

Do this after every reboot, approximately two minutes after the system has given the login prompt. Note that due to a bug in vold, CD-ROM drives may eject their media in response to this signal.

D. Sliding Controller Issue

USB 2.0 storage devices connected to a port on a USB 2.0 PCI card, and which were used with a prior Solaris OS release in the same hardware configuration, can change device names after upgrading to this release. This is because such devices are now seen as USB 2.0 devices and are taken over by the EHCI controller. (With prior Solaris OS releases, they were seen only as USB 1.1 devices and were operated through the OHCI controller.) Such devices change their controller number (the `w` in `/dev/[r]disk/cwtxdysz`).

IV.All USB Storage Devices Treated as Removable Media

This change was implemented to fix inconsistent behavior in previous releases. Some USB storage devices were treated as removable devices and others were not.

This change is implemented for all USB storage devices, connected to either framework.

1. What This Change Means

In general, USB devices will operate with a more consistent interface, and will be easier to use:

- The `rmformat(1)` program can now be used to format USB storage devices, instead of other, more complicated programs such as `format(1M)`.
- USB hard drives with a FAT file system can be mounted and used.
- Non-root users can now access USB storage devices, since a root-privileged mount command is no longer needed. The volume manager will mount these devices and make them available under `/rmdisk`. (See caveats below for restrictions.)
- All USB storage devices are power managed, except for those which support LOG SENSE pages. Devices with log sense pages are usually SCSI devices connected through a USB to SCSI bridge device.
- Error handling is more forgiving for removable media devices, with increased timeouts for recovery or preparing for the drive to become ready.

2. Caveats

- External removable media applications may not work correctly if they make assumptions about the size of the media or are not largefile aware. For example, a number field containing a disk size may overflow.
- If a new USB mass storage device is connected while the system is down, a reconfiguration boot (`boot -r`) is required for the volume manager to find the device for automatic mounting. If a new device is connected while the system is up, the volume manager must be restarted. Refer to `vold(1M)`, `scsa2usb(7D)` and the System Administration Guide for more information.
- `format(1M)` must now be started in expert mode (`-e`) in order to see any USB storage device. In previous releases, those USB storage devices which were not seen as removable media

could be seen without the `-e` switch. Users must either answer questions about device geometry or furnish an entry for `/etc/format.dat` in order to utilize the program.

- Some applications which made `DKIOCEJECT` ioctls may have returned `ENOTTY` (the error for “not applicable”) for non-removable media in the past. These calls will instead now succeed with no effect, since the device is now treated as a removable media device.
- Setting `"removable=false"` in `/kernel/drv/scsa2usb.conf` (original framework) or `/kernel/drv/usba10_scsa2usb.conf` (USBA 1.0 framework) for a particular device type overrides the forced treatment of that device type as removable media. Use this option if compatibility with previous releases is an issue. Please see these configuration files for more details.

V. USB Floppy Devices

USB floppy devices will appear as removable media devices, as all other USB disks do. USB floppy devices are not managed by the floppy `fd(7D)` driver. Applications which issue `ioctl(2)` calls intended for the `fd` driver will fail. Applications which issue only `read(2)` and `write(2)` calls will work. Some other applications, such as `SunPCI` and `rmformat`, will work.

1. Use with Volume Manager and File Manager

Volume Manager sees the USB floppy as a SCSI removable media device. Volume Manager makes the device available for access under `/rmdisk`.

File Manager does not fully support USB floppy at this time. Floppies containing a `ufs` file system can be opened, renamed and formatted from the File Manager's Removable Media Manager. Floppies containing a `pcfs` file system can only be opened from the Removable Media Manager. Drag and drop of files works with floppies containing either type of file system.

Always do a `reconfigure boot (boot -r)` when rebooting your system, if you plan on using a connected USB floppy device with Volume Manager and File Manager. When you hotplug a new USB floppy device, you must stop and restart `vold` in order for `vold` to find the newly connected device.

2. Manual configuration

This section describes how to use a USB floppy without the File Manager.

After plugging in the device, find the device's name:

```
# cd /dev/rdisk
# devfsadm -C
# ls -l c*0 | grep usb
lrwxrwxrwx  1 root  root   55 Mar  5 10:35 c2t0d0s0 ->
../../../../devices/pci@1f,0/usb@c,3/storage@3/disk@0,0:a,raw
```

Here, the device name shows as `c2t0d0s0`. The device can be accessed as raw device /

dev/rdisk/c2t0d0s0, and as block device /dev/dsk/c2t0d0s0.

If multiple device names are displayed, determine which name corresponds to the USB floppy device. Insert a floppy disk and attempt to access the device. Read the raw partition of each device (using `od`, for example) until the indicator lamp of the USB floppy lights to indicate that you have found the device.

Floppies support both `pcfs(7FS)` and `ufs` file systems. Below are examples of how the different file systems are used. The examples assume a device of `/dev/[r]dsk/c2t0d0s0`.

Use the following commands to create and use the floppy as a `pcfs` file system:

<i>Operation</i>	<i>Command</i>
Format	<code>rmformat -Flong /dev/rdisk/c2t0d0s0</code>
Create 1.4 Mb filesystem	<code>mkfs -F pcfs -o nofdisk,size=2880 /dev/rdisk/c2t0d0s0</code>
Mount	<code>mount -F pcfs /dev/dsk/c2t0d0s0:c /mnt</code>
unmount	<code>umount /mnt</code>

Use the following commands to create and use the floppy as a `ufs` file system:

<i>Operation</i>	<i>Command</i>
Format	<code>rmformat -Flong /dev/rdisk/c2t0d0s0</code>
Create 1.4 Mb filesystem	<code>newfs /dev/rdisk/c2t0d0s0</code>
Mount	<code>mount /dev/dsk/c2t0d0s0:c /mnt</code>
unmount	<code>umount /mnt</code>

VI.Driver Development Kit

Developers wishing to write either a userland or kernel USB driver for the Solaris operating system can request a Driver Development Kit (DDK). For kernel driver writers, this kit supplies full documentation of the USB 1.0 programming interface. This includes the USB 1.0 whitepaper and manpages, and also includes header files needed to compile USB 1.0 client drivers. The kit includes header files and programming examples for using the Generic USB pass-through driver, UGEN. Please go to <http://solde.sun.com/developer/support/driver/usb.html> or contact USB-DDK-help@sun.com for registration and more information.

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
650 960-1300

For U.S. Sales Office locations, call:
800 821-4643
In California:
800 821-4642

Australia: (02) 413 2666
Belgium: 32 2 716 7911
Canada: 416 477-6745
Finland: 358-0-502 27 00
France: (1) 30 67 50 00
Germany: (0) 89-46 00 8-0
Hong Kong: 852 802 4188
Italy: 039 60551
Japan: (03) 221-7021
Korea: 822-563-8700
Latin America: 415 688-9464
The Netherlands: 033 501234
New Zealand: (04) 499 2344
Nordic Countries: +46 (0) 8 623 90 00
PRC: 861-831-5568
Singapore: 224 3388
Spain: (91) 5551648
Switzerland: (1) 825 71 11
Taiwan: 2-514-0567
UK: 0276 20444

Elsewhere in the world,
call Corporate Headquarters:
650 960-1300
Intercontinental Sales: 650 688-9000